# Implementation of transfer learning to classify 10 species of Iran snakes

Iman Ahmadi*

Department of Genetics and Plant Production Engineering, Institute of Agriculture, Water, Food and Nutraceuticals, Isf. C., Islamic Azad University, Isfahan, Iran

Received: 6 May 2024 Accepted: 24 October 2024

**Key words**

Classification, EfficientNet pre-trained model, transfer learning

**Abstract**

In this study using transfer learning method, a model has been developed to classify pictures of Iran snakes in one of the 10 classes considered. Prediction accuracy of snake type and poisoning severity (in three classes named: venomous, semi-venomous, and non-venomous) using a limited dataset (composed of 174 snake pictures divided into 112 pictures of the train dataset, and 62 pictures of the test dataset) was the primary aim followed in this study. The pre-trained model that our transfer learning model was based on it was the EfficientNet model. After training the transfer learning model, it was evaluated using the sensitivity, specificity, precision, F1 score, and accuracy criteria. According to the results of this study, training phase of the model lasted about 60 minutes and its accuracy was 76%. Due to the fact that the model was trained on a regular laptop computer without a GPU, its performance was acceptable.

*Corresponding Author: imanahmadi1358@iau.ac.ir

# پیاده‌سازی یادگیری انتقالی در شناسایی ۱۰ گونه از مارهای ایران

**ایمان احمدی***

گروه مهندسی تولید و ژنتیک گیاهی، دانشکده کشاورزی، آب، غذا و فراسودمندها، واحد اصفهان (خوراسگان)، دانشگاه آزاد اسلامی، اصفهان، ایران

دریافت: ۱۷ اردیبهشت ۱۴۰۳ پذیرش: ۳ آبان ۱۴۰۳

**واژه‌های کلیدی**

دسته‌بندی، ماتریس درهم ریختگی، مدل EfficientNet، یادگیری انتقالی

**چکیده**

در این پژوهش با روش یادگیری انتقالی که یکی از الگوریتمهای هوش مصنوعی در دسته‌بندی تصاویر است، مدلی توسعه داده شده است که توسط آن دسته‌بندی تصاویر مارهای غالب ایران در یکی از ۱۰ دسته مورد نظر صورت گیرد. هدف این است که توانایی مدلِ برمبنای یادگیری انتقالی در پیش‌بینی گونه مار و شدت سمی بودن آن (در سه سطح سمی، نیمه سمی و غیر سمی) با استفاده از تعداد محدودی تصویر از مارهای ایران که شامل ۱۷۴ تصویر بوده و به ۱۱۲ تصویر آموزشی اولیه (۱۱۲۰ تصویر آموزشی نهایی با عمل افزایش نرم‌افزاری تعداد تصاویر) و ۶۲ تصویر آزمونی تفکیک شدند، بررسی گردد. مدل از پیش آموزش دیده‌ای که روش یادگیری انتقالی مورد نظر در این پژوهش بر مبنای آن استوار است، مدل EfficientNet بوده و پس از آموزش، با محاسبه مقادیر معیارهای حساسیت، اختصاصیت، دقت، نمره F1 و صحت، کارایی مدلِ برمبنای یادگیری انتقالی ارزیابی شد. بر طبق نتایج به دست آمده، زمان آموزش مدل حدود ۶۰ دقیقه و صحت مدل ۷۶ درصد به‌دست آمد. از آنجا که مدل حاضر روی رایانه بدون GPU توسعه یافته است، کارکرد مدل توسعه یافته قابل قبول می‌باشد.

* پست الکترونیکی: imanahmadi1358@iau.ac.ir

**مقدمه**

مارها نوعی خزنده هستند که وجودشان برای حفظ سلامت اکوسیستم لازم است. هنگامی که انسانها با این جانور مواجه می‌شوند معمولاً در صدد گرفتن و یا کشتن آن برمی‌آیند که این کار می‌تواند به شدت خطرناک باشد. از آنجا که بسیاری از انسانها اطلاعی درباره گونه‌های مختلف مارها ندارند، تصور می‌کنند که تمام مارها سمی هستند.

از طرف دیگر اگر گونه این جانوران با چشم تشخیص داده شود، شناسایی غلط ممکن است صورت گیرد که نتایج مرگباری را می‌تواند به همراه داشته باشد. همچنین شناسایی گونه مار برای انجام تیمارهای درمانی صحیح برای اشخاص نیش زده شده توسط این جانوران و بازگشت سلامتی به این افراد ضروری است. این شناسایی می‌تواند به صورت چشمی و به کمک ویژگی‌های مار مانند خط و خال مار، شکل سر، رنگ پوست، شکل بدن و چشم و ... انجام شود. اما افراد غیر حرفه‌ای با این ویژگیها به طور کامل آشنا نیستند که این امر باعث ایجاد تأخیر در شناسایی صحیح مار می‌شود که ممکن است به مرگ بیمار منجر شود (Picek et al., 2022). اولین گام در حفاظت از گونه‌های در معرض خطر انقراض نیز فراهم کردن امکان تشخیص این گونه‌ها است.

در نقطه مقابلِ روش چشمی، سامانه های تشخیص گونه مار به کمک پردازش تصویر به منظور دسته بندی مارها با استفاده از روشهای یادگیری ماشین توسعه یافته‌اند که در این روشها، ویژگیها و پارامترهای استفاده شده برای دسته بندی، توسط افراد خبره تنظیم می‌شود. امروزه با دسترسی به حجم زیاد داده و تصویر، سرو کار داشتن با داده‌های حجیم غیر قابل اجتناب است، اما روشهای یادگیری ماشین قادر به مدیریت این حجم داده نمی‌باشند. بنابراین روند به سمت استفاده از روشی که یادگیری عمیق نام گرفته، متمایل شده که با این روش، مدیریت حجم زیاد داده قابل انجام است. مهمترین امتیاز روشهای یادگیری عمیق این است که در این روشها ویژگیهای تصویر، توسط الگوریتم رایانه‌ای کشف می‌شود، بنابراین نیاز به فرد خبره برای تنظیم ویژگیها مرتفع می‌شود. هنگامی‌که روشهای یادگیری عمیق به‌کار گرفته می‌شود، دسته‌بندی آسانتر و سریعتر صورت می‌گیرد. در زیر به برخی از پژوهشهای انجام شده توسط محققین با رویکرد یادگیری ماشین و یادگیری عمیق در شناسایی گونه‌های مار پرداخته شده است: Amir et al., (2017) با استفاده از ۵ روش یادگیری ماشین به دسته‌بندی ۳۴۹ تصویر از مارهای مالزی در ۲۲ کلاس اقدام کرده و صحتهای به‌دست آمده در این تحقیق در بازه ۷۰٪ تا ۸۰٪ قرار گرفتند. James (2017) از روش KNN برای دسته‌بندی ۴۴۴ تصویر مار در ۶ کلاس استفاده کرد و صحت ۸۵٪ را به‌دست آورد. استفاده از روش شبکه عصبی کانولوشنال (CNN) نیز در برخی از پژوهشها برای دسته‌بندی تصاویر مار مورد استفاده قرار گرفته است و صحتهای به‌دست آمده اغلب در بازه ۸۰٪ تا ۹۵٪ قرار داشتند ( Abdurrazaq et al., (2019); Abayaratne et al., (2019); Progga et al., (2021); Yang & Sinnott (2021); Kalinathan et al., (2021)). در تعداد محدودی از پژوهشها از روش یادگیری انتقالی برای دسته‌بندی تصاویر مارها استفاده شده است (Desingu et al., (2021)) اما صحت پیش بینی انجام گرفته اندک و در حد ۴۲٪ بوده است.

در این پژوهش با روش یادگیری انتقالی، مدلی توسعه داده شده است که توسط آن دسته‌بندی تصاویر مارهای غالب ایران در یکی از ۱۰ دسته مورد نظر صورت گیرد. هدف این است که توانایی مدل برمبنای یادگیری انتقالی در پیش‌بینی گونه مار و شدت سمی بودن آن با استفاده از تعداد محدودی تصویر از مارهای عمده ایران بررسی گردد. ذکر این نکته اهمیت دارد که این مدل روی یک رایانه مجهز به CPU معمولی و فاقد GPU اجرا شده است.

## مواد و روشها

### آماده سازی تصاویر خام

هدف از این مقاله تشخیص ۱۰ گونه از مارهای غالب موجود در ایران است. نامهای علمی و فارسی این ۱۰ گونه مار در جدول ۱ آورده شده است. از نام علمی این مارها به همراه شدت سمیت آنها به عنوان نام گروه‌های دسته‌بند استفاده شد. در این پژوهش سمیت مارها در سه کلاس غیر سمی (non-venomous)، نیمه سمی (semi-venomous) و سمی (venomous) تقسیم بندی شده است.

**جدول ۱. نامهای علمی و فارسی به همراه شدت سمیت گونه‌های مار مورد استفاده در این پژوهش (نامهای علمی گونه‌های مار به همراه شدت سمیت آنها به عنوان نامهای گروه‌های دسته‌بند در این پژوهش مورد استفاده قرار گرفت)**

| نام علمی | نام فارسی | شدت سمیت |
|---|---|---|
| *Echis_carinatus* | مار جعفری | سمی |
| *Eryx_jaculus* | کور مار | غیر سمی |
| *Gloydius_halys* | افعی قفقازی | سمی |
| *Macrovipera_lebetinus* | گرزه مار | سمی |
| *Malpolon_moilensis* | طلحه مار | نیمه سمی |
| *Naja_oxiana* | کفچه مار | سمی |
| *Natrix_natrix* | مار آبی | غیر سمی |
| *Platyceps_karelini* | مار خالدار | غیر سمی |
| *Psammophis_schokari* | تیر مار | نیمه سمی |
| *Spalerosophis_diadema* | شتر مار | غیر سمی |

تعدادی تصویر متعلق به هر یک از گروه‌های دسته بند برای توسعه مدل مورد استفاده قرار گرفت. تعداد کل تصاویر مورد استفاده در این پژوهش برابر با ۱۷۴ تصویر بود. ابتدا تصاویر هر گروه دسته‌بند با نسبت تقریبی ۲ به ۱ به دو زیر گروه تقسیم شدند. سپس برای هر یک از ۱۰ زیر گروه، تصاویر زیر گروه بزرگتر به یک پوشه دارای نام یکسان با نام دسته‌بند مربوطه منتقل شدند و همه این پوشه‌ها در پوشه دیگری تحت عنوان "train" کپی شدند (کل تصاویر اولیه پوشه train برابر با ۱۱۲ تصویر بود). روند مشابهی برای هر یک از زیر گروه‌های کوچکتر تقسیم‌بندی ۱۰ گانه فوق صورت گرفت، با این تفاوت که تمام پوشه‌های به‌دست آمده در پوشه دیگری تحت عنوان "test" کپی شدند (کل تصاویر اولیه پوشه train برابر با ۶۲ تصویر بود). همچنین برای افزایش تعداد تصاویر آموزشی از روش image augmentation در این پژوهش استفاده شد و تعداد تصاویر آموزشی ۱۰ برابر گردید. سپس پوشه‌های "train" و "test" به پوشه نهایی تحت عنوان "Iran_Snakes" انتقال یافتند. این پوشه در مسیر کاری زبان برنامه‌نویسی Python به‌منظور توسعه مدل کپی گردید. تعداد کل تصاویر نهایی قرار گرفته در پوشه‌های "train" و "test" بترتیب برابر با ۱۱۲۰ و ۶۲ تصویر بود. تصاویر موجود در پوشه "train" برای آموزش مدل استفاده شد (منظور از آموزش مدل، تغییر وزنهای مدل در جهتی است که پیش‌بینی‌های مدل روی تصاویر "train" با واقعیت انطباق یابد)، در حالیکه تصاویر موجود در پوشه "test" برای آزمایش خوب بودن وزنهای به دست آمده برای تشخیص گونه مارهای موجود در تصاویر دیده نشده پوشه "test" به کار برده شد.

تمام تصاویر مورد استفاده به همراه پیش‌بینی‌های مدل روی تصاویر پوشه test، از طریق آدرس فضای ابری گوگل درایو زیر قابل دستیابی است.

https://drive.google.com/file/d/1I_E2zLjg7W_1O6NI-t537DiTn3VWKE8r/view?usp=sharing

در صورت مطالعه مطالب چاپ شده روی کاغذ، با اسکن QRcode زیر هم می‌توانید به تصاویر دسترسی داشته باشید.

## روند توسعه مدل

توسعه مدل با کد نوشته شده در محیط برنامه‌نویسی PyCharm با استفاده از کتابخانه PyTorch زبان برنامه‌نویسی Python انجام شد. کد مربوط به مدل به انتهای مقاله پیوست شده است. در کتابخانه PyTorch توابعی وجود دارد که این کتابخانه را برای ایجاد مدلهای برمبنای مفهوم یادگیری انتقالی مناسب می‌کند. به عنوان مثال وزنهای مدل آموزش داده شده از قبل (مانند مدل EfficientNet) با کد زیر قابل دستیابی است:

```
weights = torchvision.models.EfficientNet_B_Weights.DEFAULT
```

از سوی دیگر، از آنجا که آماده‌سازی تصاویر جدیدی که وارد مدلِ برمبنای یادگیری انتقالی می‌شوند باید مشابه آماده‌سازیِ تصاویر اولیه وارد شده به مدلِ آموزش داده شده از قبل (EfficientNet) باشد، استفاده از کد زیر باعث می‌شود که کتابخانه PyTorch تمام تبدیلات داده‌ای انجام شده برای آموزش وزنها در مدل EffivientNet را به صورت خودکار به دست آورد:

```
auto_transforms = weights.transformers()
```

به منظور کارکرد صحیح مدل نهایی بر مبنای مفهوم یادگیری انتقالی این تبدیلات باید روی داده‌های جدید ورودی به مدل اعمال شوند. به علت وجود این توابع در کتابخانه PyTorch، استفاده از آن برای مدلسازان یادگیری عمیق جذاب است. معماری مدل برمبنای یادگیری انتقالی (TL) برای دسته‌بندی ۱۰ گونه از مارهای غالب ایران در جدول ۲ نشان داده شده است:

همانطور که مشاهده می‌شود، پارامترهای بخش استخراج ویژگیها [Sequential (features)] در طول فرآیند آموزش مدل امکان تغییر ندارند (قابلیت آموزش دیدن پارامترهای بخش استخراج ویژگیها False لحاظ شده است)، و فقط پارامترهای مربوط به بخش دسته‌بند [Sequential (classifier)] قابل تغییر و آموزش هستند. این کار باعث کاهش قابل توجه در زمان آموزش مدل می‌شود. از نظر عددی، تعداد پارامترهای قابل آموزش از ۴۰۲۰۳۵۸ پارامتر مربوط به مدل EfficientNet به ۱۲۸۱۰ پارامتر مربوط به مدل برمبنای مفهوم یادگیری انتقالی (TL) کاهش یافته است. این ویژگی مدل TL باعث می‌شود که امکان اجرای این مدل روی رایانه مجهز به CPU معمولی در زمان منطقی فراهم گردد.
مقادیر پارامترهای مدل TL در جدول ۳ آورده شده است:

**جدول ۲. معماری مدل یادگیری انتقالی (TL) استوار بر مدل از پیش آموزش داده شده EfficientNet**

| Layer (type (var_name)) | Input Shape | Output Shape | Param # | Trainable |
|---|---|---|---|---|
| EfficientNet (EfficientNet) | [32, 3, 224, 224] | [32, 10] | -- | Partial |
| ├─Sequential (features) | [32, 3, 224, 224] | [32, 1280, 7, 7] | -- | False |
| │ └─Conv2dNormActivation (0) | [32, 3, 224, 224] | [32, 32, 112, 112] | -- | False |
| │ │ └─Conv2d (0) | [32, 3, 224, 224] | [32, 32, 112, 112] | (864) | False |
| │ │ └─BatchNorm2d (1) | [32, 32, 112, 112] | [32, 32, 112, 112] | (64) | False |
| │ │ └─SiLU (2) | [32, 32, 112, 112] | [32, 32, 112, 112] | -- | -- |
| │ └─Sequential (1) | [32, 32, 112, 112] | [32, 16, 112, 112] | -- | False |
| │ │ └─MBConv (0) | [32, 32, 112, 112] | [32, 16, 112, 112] | (1,448) | False |
| │ └─Sequential (2) | [32, 16, 112, 112] | [32, 24, 56, 56] | -- | False |
| │ │ └─MBConv (0) | [32, 16, 112, 112] | [32, 24, 56, 56] | (6,004) | False |
| │ │ └─MBConv (1) | [32, 24, 56, 56] | [32, 24, 56, 56] | (10,710) | False |
| │ └─Sequential (3) | [32, 24, 56, 56] | [32, 40, 28, 28] | -- | False |
| │ │ └─MBConv (0) | [32, 24, 56, 56] | [32, 40, 28, 28] | (15,350) | False |
| │ │ └─MBConv (1) | [32, 40, 28, 28] | [32, 40, 28, 28] | (31,290) | False |
| │ └─Sequential (4) | [32, 40, 28, 28] | [32, 80, 14, 14] | -- | False |
| │ │ └─MBConv (0) | [32, 40, 28, 28] | [32, 80, 14, 14] | (37,130) | False |
| │ │ └─MBConv (1) | [32, 80, 14, 14] | [32, 80, 14, 14] | (102,900) | False |
| │ │ └─MBConv (2) | [32, 80, 14, 14] | [32, 80, 14, 14] | (102,900) | False |
| │ └─Sequential (5) | [32, 80, 14, 14] | [32, 112, 14, 14] | -- | False |
| │ │ └─MBConv (0) | [32, 80, 14, 14] | [32, 112, 14, 14] | (126,004) | False |
| │ │ └─MBConv (1) | [32, 112, 14, 14] | [32, 112, 14, 14] | (208,572) | False |
| │ │ └─MBConv (2) | [32, 112, 14, 14] | [32, 112, 14, 14] | (208,572) | False |
| │ └─Sequential (6) | [32, 112, 14, 14] | [32, 192, 7, 7] | -- | False |
| │ │ └─MBConv (0) | [32, 112, 14, 14] | [32, 192, 7, 7] | (262,492) | False |
| │ │ └─MBConv (1) | [32, 192, 7, 7] | [32, 192, 7, 7] | (587,952) | False |
| │ │ └─MBConv (2) | [32, 192, 7, 7] | [32, 192, 7, 7] | (587,952) | False |
| │ │ └─MBConv (3) | [32, 192, 7, 7] | [32, 192, 7, 7] | (587,952 | False |
| │ └─Sequential (7) | [32, 192, 7, 7] | [32, 320, 7, 7] | -- | False |
| │ │ └─MBConv (0) | [32, 192, 7, 7] | [32, 320, 7, 7] | (717,232) | False |
| │ └─Conv2dNormActivation (8) | [32, 320, 7, 7] | [32, 1280, 7, 7] | -- | False |
| │ │ └─Conv2d (0) | [32, 320, 7, 7] | [32, 1280, 7, 7] | (409,600) | False |
| │ │ └─BatchNorm2d (1) | [32, 1280, 7, 7] | [32, 1280, 7, 7] | (2,560) | False |
| │ │ └─SiLU (2) | [32, 1280, 7, 7] | [32, 1280, 7, 7] | -- | -- |
| ├─AdaptiveAvgPool2d (avgpool) | [32, 1280, 7, 7] | [32, 1280, 1, 1] | -- | -- |
| ├─Sequential (classifier) | [32, 1280] | [32, 10] | -- | True |
| │ └─Dropout (0) | [32, 1280] | [32, 1280] | -- | -- |
| │ └─Linear (1) | [32, 1280] | [32, 10] | 12,810 | True |

Total params: 4,020,358
Trainable params: 12,810
Non-trainable params: 4,007,548
Total mult-adds (Units.GIGABYTES): 12.31

Input size (MB): 19.27
Forward/backward pass size (MB): 3452.09
Params size (MB): 16.08
Estimated Total Size (MB): 3487.44

**جدول ۳. مقادیر پارامترهای مدل TL**

| نام انگلیسی | نام فارسی | مقدار |
|---|---|---|
| Training epoch | تعداد دوره‌های آموزش | ۲۵ |
| Batch size | تعداد تصاویر مربوط به گروه‌های تصویری استفاده شده در آموزش مدل | ۳۲ |
| optimizer | تابع بهینه‌ساز | Adam |
| Learning rate | نرخ یادگیری | ۰,۰۰۱ |
| Loss function | تابع هزینه | CrossEntropy |

## معیارهای ارزیابی مدل

### زمان آموزش

مدل دارای زمان آموزش کم از مدل دارای زمان آموزش زیاد بهتر است. بنابراین زمان آموزش به عنوان یکی از معیارهای ارزیابی در نظر گرفته شد. اهمیت زمان آموزش در مدلهای یادگیری عمیق در حال اجرا روی رایانه‌های مجهز به CPU معمولی به جای GPU بالاتر می‌رود.

### حساسیت Sensitivity (Recall)

این معیار خوب بودن مدل در شناسایی نمونه‌های مثبتِ صحیح را می‌رساند. به زبان ریاضی حساسیت با استفاده از فرمول زیر قابل محاسبه است:

$$\text{Sensitivity (Recall)} = \frac{\text{TP}}{\text{TP+FN}} \times 100$$

جاییکه TP تعداد نمونه‌هایی است که به طور صحیح، مثبت پیش بینی شده‌اند و FN تعداد نمونه‌هایی است که به اشتباه، منفی پیش بینی شده‌اند. به بیان دیگر TP تعداد تصاویر متعلق به یک دسته خاص مار است که مدل هم آنها را متعلق به آن دسته خاص پیش بینی کرده است و FN تعداد تصاویر متعلق به یک دسته خاص مار است که مدل آنها را به اشتباه متعلق به دسته‌های دیگر پیش بینی کرده است. در واقع مخرج کسر تعداد تصاویری که به طور واقعی متعلق به آن دسته خاص هستند را می‌رساند و صورت کسر تعداد تصاویر درست پیش بینی شده متعلق به آن دسته خاص را مشخص می‌کند.

### اختصاصیت Specificity

این معیار توانایی مدل در شناسایی نمونه‌های منفیِ صحیح هر گروه دسته‌بند را می‌رساند. به زبان ریاضی اختصاصیت با استفاده از فرمول $\text{Specificity} = \frac{\text{TN}}{\text{TN+FP}} \times 100$ قابل محاسبه است، جاییکه TN تعداد نمونه‌هایی است که به طور صحیح، منفی پیش بینی شده‌اند و FP تعداد نمونه‌هایی است که به اشتباه، مثبت پیش بینی شده‌اند.

به بیان دیگر اگر دسته خاصی از مارها در نظر گرفته شود، TN تعداد تصاویر متعلق به سایر دسته‌های مار است که مدل هم آنها را متعلق به آن دسته خاص پیش بینی نکرده است و FP تعداد تصاویر متعلق به سایر دسته‌های مار است که مدل آنها را به اشتباه متعلق به آن دسته خاص پیش بینی کرده است. در واقع مخرج کسر تعداد تصاویری که به طور واقعی متعلق به آن دسته خاص نیستند را می‌رساند و صورت کسر تعداد تصاویری را نشان می‌دهد که به طور صحیح عدم تعلق آنها به آن دسته خاص پیش‌بینی شده است.

### دقت Precision

این معیار نسبت تعداد نمونه‌هایی که به طور صحیح مثبت پیش‌بینی شده‌اند به تعداد کل نمونه‌هایی که مثبت پیش‌بینی شده‌اند را نشان می‌دهد. به زبان ریاضی دقت با استفاده از فرمول زیر قابل محاسبه است: $\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \times 100$.

### نمره F1 یا F1 score

نمره F1 با محاسبه میانگین هارمونیک حساسیت و دقت به‌دست می‌آید. به بیان ریاضی داریم: $F1\,score = \frac{2 \times Precision \times Recall}{Precision+Recall} \times 100$. هر گاه یکی از مقادیر حساسیت و یا دقت صفر باشند، نمره F1 صفر می‌شود، درحالیکه میانگین حسابی این دو پارامتر عددی بزرگتر از صفر را نتیجه خواهد داد.

### صحت Accuracy

این معیار نسبت تعداد نمونه‌های به طور صحیح پیش‌بینی شده توسط مدل به تعداد کل پیش‌بینی‌ها را نشان می‌دهد. به بیان ریاضی داریم: $Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \times 100$. در طول فرآیند آموزش، این پارامتر پس از هر دوره آموزش(epoch) محاسبه می‌شود (در این مطالعه تعداد دوره‌های آموزش ۲۵ دوره در نظر گرفته شد)، سپس مقادیر صحت محاسبه شده برحسب عدد دوره رسم می‌شوند. نمودار به‌دست آمده برای اطمینان یافتن از کفایت تعداد دوره‌های آموزشی به کار می‌رود، به بیان دیگر اگر نمودار صحت برحسب عدد دوره آموزشی در طول تعدادی دوره (مثلاً ۵ دوره) افقی باقی بماند، این موضوع نشان دهنده کفایت تعداد دوره آموزشی برای آموزش مدل است.

### ماتریس در هم ریختگی Confusion matrix

برای به‌دست آمدن شناخت کاملی از تعداد پیش‌بینی‌های غلط مدل و اینکه گروهِ صحیح با کدام گروه‌ها قاتی شده‌است، از ماتریس در هم ریختگی استفاده می‌شود. در این مطالعه یک ماتریس در هم ریختگی ۱۰×۱۰ برای مشخص شدن مکان پیش‌بینی‌های غلط ۱۰ گونه مار توسعه یافته است. در یک ماتریس در هم ریختگی، هر چه تمرکز مقادیر بزرگ روی قطر اصلی ماتریس بیشتر باشد (یعنی مقادیر صفر و کوچک بیشتری در سایر سلولهای ماتریس باشد) بهتر است.

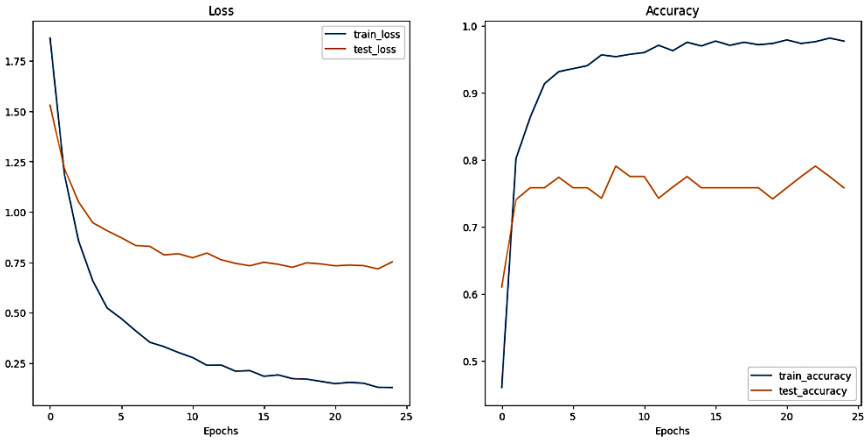
## نتایج

### زمان آموزش مدل:

زمان کل مصرف شده برای آموزش مدل ۳۶۵۱ ثانیه (حدود ۶۰ دقیقه) شد. از آنجا که مدل روی رایانه قابل حملی که مجهز به CPU معمولی بود آموزش داده شده است، این زمان منطقی است؛ هر چند در صورت استفاده از رایانه مجهز به GPU برای آموزش مدل، این زمان به مقدار قابل توجهی کاهش می‌یابد.

### مقادیر صحت برحسب شماره دوره:

شکل ۱ صحتها و مقدار تابع هزینه به دست آمده در طول فرآیندهای آموزش و آزمون برحسب شماره دوره را نشان می‌دهد:



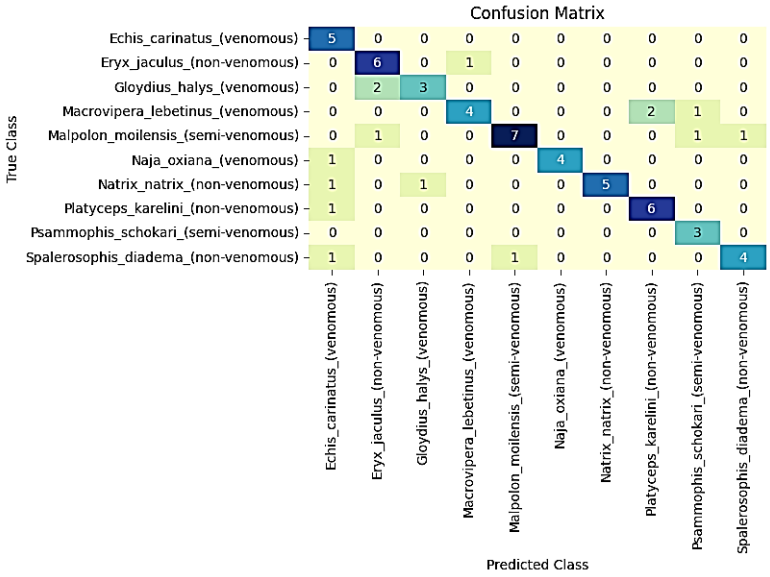

**شکل ۱. صحتهای فرآیندهای آموزش و آزمون برحسب شماره دوره**

اولین نکته‌ای که از شکل ۱ به دست می‌آید این است که در شماره دوره بیش از ۱۵، نوسانهای مقادیر صحت و تابع هزینه در حد ۱۰٪ بوده و صحت مدل در بخش آزمون به حدود ۸۰٪ رسیده است، در نتیجه مدل خوب آموزش دیده است و افزایش شماره دوره به بیش از ۲۵ باعث افزایش چشمگیری دیگری در صحت بخش آزمون نخواهد شد. از طرف دیگر از آنجا که فاصله بین صحتهای بخش آموزش و آزمون در شماره دوره‌های بالای ۱۵ به حدود ۲۰٪ رسیده و این وضعیت در صحتهای آموزش نزدیک ۱۰۰٪ اتفاق افتاده، در نتیجه برای مدل وضعیت بیش برازشی[1] و کم برازشی[2] پیش نیامده است.

### ماتریس درهم ریختگی برای نمونه‌های آزمون:

شکل ۲ ماتریس درهم ریختگی به‌دست آمده از ۶۲ تصویر متعلق به مجموعه داده‌ای آزمون را نشان می‌دهد:

1 - overfitting
2 - underfitting




**شکل ۲. ماتریس درهم ریختگی به‌دست آمده در این مطالعه**

این شکل ماتریس برچسبهای واقعی در برابر برچسبهای پیش‌بینی شده توسط مدل را برای تصاویر آزمون نشان می‌دهد و اعداد موجود در سلولهای ماتریس بیانگر تعداد تصاویر متعلق به هر ترکیب از برچسبهاست. همانطور که مشاهده می‌شود اعداد ثبت شده در سلولهای قرار گرفته در قطر اصلی این ماتریس بزرگتر از اعداد ثبت شده در سایر سلولهای ماتریس است. اعداد موجود در سلولهای قطر اصلی ماتریس تعداد تصاویر متعلق به مجموعه داده‌ای آزمون مربوط به هر گروه از گروه‌های دسته‌بند را نشان می‌دهد که مدل برچسب آنها را به‌درستی مثبت پیش‌بینی کرده است. به بیان دیگر این مقادیر تعداد تصاویر TP مربوط به هر گروه از گروه‌های ۱۰گانه مارهای عمده ایران را نشان می‌دهد. تعداد تصاویر FP و FN و TN متعلق به هر یک از گروه‌های دسته‌بند نیز به راحتی به کمک این ماتریس قابل محاسبه است. در هر ردیف از ماتریس، مجموع اعداد سلولها به جز سلول قرار گرفته در قطر اصلی ماتریس، تعداد تصاویر FN متعلق به گروه دسته‌بند مربوطه را نشان می‌دهد. همچنین در هر ستون از ماتریس، مجموع اعداد سلولها به جز سلول قرار گرفته در قطر اصلی ماتریس، تعداد تصاویر FP متعلق به گروه دسته‌بند مربوطه را نشان می‌دهد و در نهایت تعداد تصاویر TN مربوط به هر گروه از گروه‌های دسته‌بند از رابطه زیر قابل محاسبه است:

$$TN = 62 - (TP + FN + FP)$$

جدول ۴ مقادیر TP، TN، FP و FN هر گروه از گروه‌های دسته‌بند مارهای غالب مورد توجه در این پژوهش را نشان می‌دهد:

**جدول ٤. اطلاعات مفیدی که از ماتریس درهم ریختگی قابل استخراج است**

| نام گروه دسته بند | سمیت مار | TN | FP | FN | TP |
|---|---|---|---|---|---|
| مار جعفری | سمی | ۵۳ | ۴ | ۰ | ۵ |
| کور مار | غیر سمی | ۵۲ | ۳ | ۱ | ۶ |
| افعی قفقازی | سمی | ۵۶ | ۱ | ۲ | ۳ |
| گرزه مار | سمی | ۵۴ | ۱ | ۳ | ۴ |
| طلحه مار | نیمه سمی | ۵۱ | ۱ | ۳ | ۷ |
| کفچه مار | سمی | ۵۷ | ۰ | ۱ | ۴ |
| مار آبی | غیر سمی | ۵۵ | ۰ | ۲ | ۵ |
| مار خالدار | غیر سمی | ۵۳ | ۲ | ۱ | ۶ |
| تیر مار | نیمه سمی | ۵۷ | ۲ | ۰ | ۳ |
| شتر مار | غیر سمی | ۵۵ | ۱ | ۰ | ۴ |

## محاسبه حساسیت، اختصاصیت، دقت، نمره F1 و صحت مدل:

مقادیر نشان داده شده در جدول ۴ برای محاسبه معیارهای حساسیت، اختصاصیت، دقت، نمره F1 و صحت مدل توسعه یافته در این پژوهش استفاده می‌شوند. در جدول ۵ مقادیر محاسبه شده معیارهای بالا نشان داده شده است:

**جدول ۵. مقادیر حساسیت، اختصاصیت، دقت و نمره F1 محاسبه شده برای گروه‌های دسته‌بند**

| نام گروه دسته بند | حساسیت | اختصاصیت | دقت | نمره F1 |
|---|---|---|---|---|
| مار جعفری | ۱۰۰ | ۹۳ | ۵۶ | ۷۲ |
| کور مار | ۸۶ | ۹۵ | ۶۷ | ۷۵ |
| افعی قفقازی | ۶۰ | ۹۸ | ۷۵ | ۶۷ |
| گرزه مار | ۵۷ | ۹۸ | ۸۰ | ۶۷ |
| طلحه مار | ۷۰ | ۹۸ | ۸۸ | ۷۸ |
| کفچه مار | ۸۰ | ۱۰۰ | ۱۰۰ | ۸۹ |
| مار آبی | ۷۱ | ۱۰۰ | ۱۰۰ | ۸۳ |
| مار خالدار | ۸۶ | ۹۶ | ۷۵ | ۸۰ |
| تیر مار | ۱۰۰ | ۹۷ | ۶۰ | ۷۵ |
| شتر مار | ۶۷ | ۹۸ | ۸۰ | ۷۳ |

مقادیر معیارهای نشان داده شده در جدول ۵ برای هر کدام از ۱۰ گروه مارهای غالب ایران محاسبه شده است. برای محاسبه معیارهای نظیر مربوط به مدل، می‌توان از میانگین‌گیری معمولی و یا وزن دار استفاده کرد. جدول ۶ مقادیر معیارهای بالا را که برای مدل به دو روش میانگین معمولی و میانگین وزن دار محاسبه شده است را نشان می‌دهد:

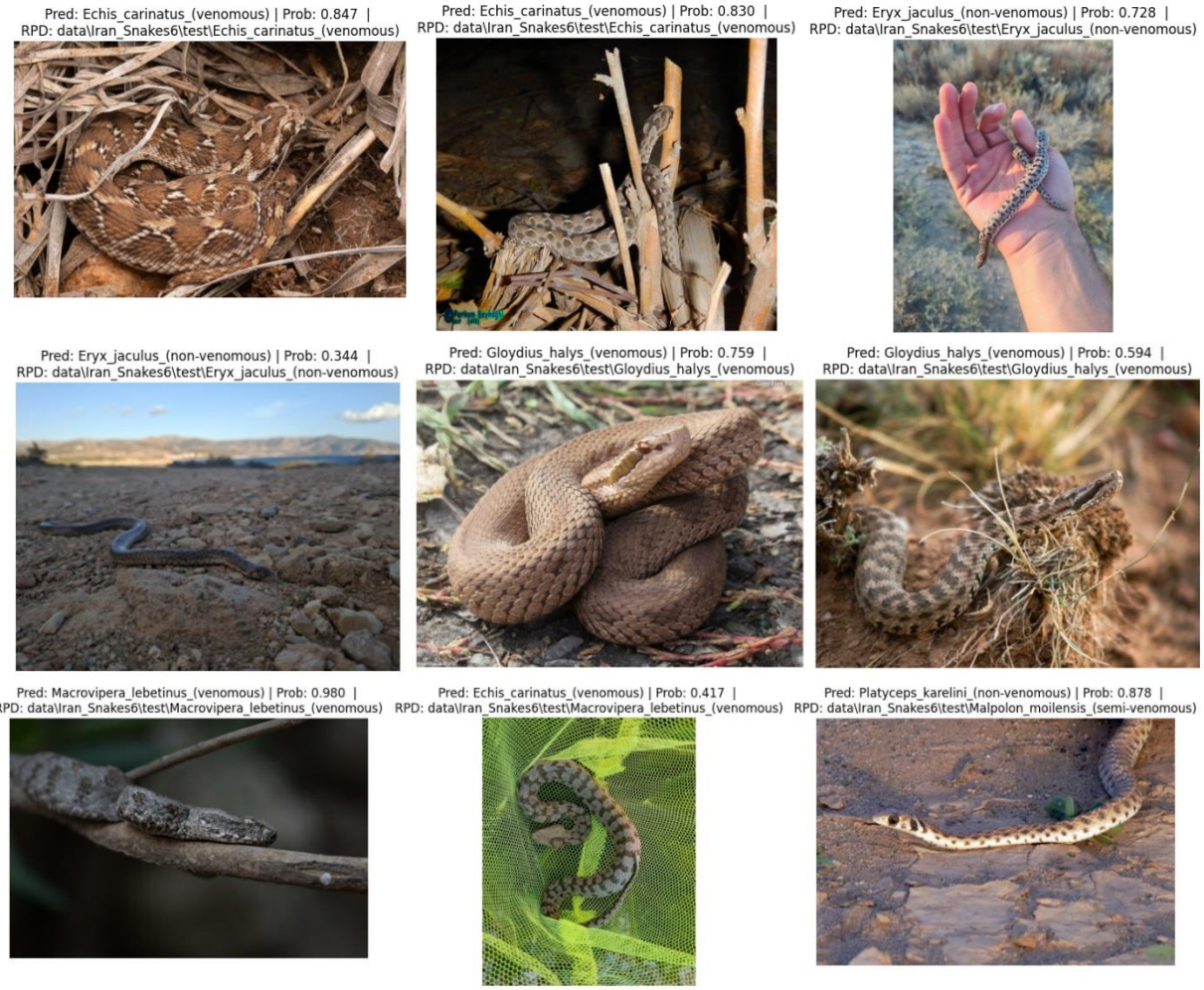
**جدول ٦. مقادیر معیارهای ارزیابی محاسبه شده برای مدل با میانگین گیری معمولی و وزن‌دار**

| روش محاسبه | حساسیت | اختصاصیت | دقت | نمره F1 |
|---|---|---|---|---|
| میانگین معمولی | ۷۷/۷ | ۹۷/۳ | ۷۸/۱ | ۷۵/۹ |
| میانگین وزن‌دار | ۷۵/۸ | ۹۷/۴ | ۷۹/۸ | ۷۶/۱ |

صحت مدل با استفاده از فرمول زیر به‌دست آمد:

$$accuracy = \frac{\sum_{i=1}^{10}(TP_i)}{\sum_{i=1}^{10}(TP_i+FP_i)} = \frac{47}{62} = 76$$

برای ارزیابی عملکرد مدل به صورت دیداری، پیش‌بینی‌های مدل روی تعدادی از تصاویر مار از گروه‌های مختلف دسته‌بند مورد توجه قرار گرفت. شکل ۳ پیش‌بینی‌های مدل روی ۲۱ تصویر انتخاب شده به صورت تصادفی از مجموعه داده‌ای آزمون را نشان می‌دهد:




**شکل ۳. پیش‌بینی‌های مدل روی تعداد تصویر مار گرفته شده از مجموعه داده‌ای آزمون**

سه واژه مخفف در بالای هر تصویر وجود دارد که عبارتند از Pred، Prob و RPD. Pred مخفف Prediction است و گروه پیش‌بینی شده مربوط به آن تصویر را نشان می‌دهد. RPD مخفف Real Parent Directory است و مسیر صحیحی که آن تصویر به آن متعلق است را مشخص می‌کند. Prob مخفف Probability است و مقدار احتمالی را نشان می‌دهد که با آن احتمال تصویر به گروه پیش‌بینی شده نسبت داده می‌شود. به بیان دیگر، الگوریتم دسته‌بند ۱۰ احتمال را برای هر تصویر محاسبه می‌کند که این احتمالات تعلق آن تصویر به هر کدام از ۱۰ گونه از مارهای غالب ایران را نشان می‌دهد و گونه دارای بزرگترین احتمال را به عنوان دسته پیش‌بینی شده برای آن تصویر بر می‌گرداند.

همانطور که دیده می‌شود ۵ تصویر از ۲۱ تصویر به طور اشتباه پیش‌بینی شده است در نتیجه نرخ خطا ۲۴٪ (یعنی نرخ صحت ۷۶٪) است..

## بحث و نتیجه‌گیری

مصالحه‌ای بین عملکرد هر یک از روشهای یادگیری ماشین و زمان و یا منابع محاسباتی اختصاص یافته به آموزش مدل باید برقرار شود. هر چند زمان اختصاص یافته برای آموزش مدل

توسعه یافته در این پژوهش حدود ۶۰ دقیقه است که نسبتاً بالاست، اما این مدل روی رایانه‌ای که مجهز به GPU نبوده است آموزش دیده است. از سوی دیگر صحت مدل توسعه یافته در این پژوهش در حد صحتهای به‌دست آمده توسط بعضی از پژوهشگران روی دسته‌بندی مارها بوده است (Abdurrazaq et al., (2019); James (2017); Amir et al., (2017)) وحتی از بعضی از صحتهای به دست آمده توسط سایر محققین که از روش TL استفاده کرده‌اند، بیشتر بوده است ((Desingu et al., (2021)؛ هر چند پژوهشهایی هم وجود داشته که در آن صحتهای بالاتری به‌دست آمده است (Abayaratne et al., (2019); Progga et al., (2021)). برای توجیه این مطلب می‌توان به دو نکته اشاره کرد: ۱- در پژوهشهایی که به صحتهای بالاتر منجر شده از روش یادگیری عمیق CNN استفاده شده است، نه از روش TL. به بیان دیگر در هنگام آموزش مدل پارامترهای بخش استخراج ویژگیها و پارامترهای بخش دسته‌بند قابل تغییر بودند در حالیکه در مدل این پژوهش فقط پارامترهای بخش دسته‌بند قابل آموزش و تغییر بودند. البته برای آموزش مدل CNN به منابع محاسباتی بیشتر احتیاج است که از این نظر روش TL بر CNN برتری دارد. ۲- در پژوهشهای دارای صحت بیشتر، تصاویر بیشتری مورد استفاده قرار گرفته است، در حالیکه مجموعه تصاویر این پژوهش محدود به ۱۷۴ تصویر بوده است. بنابراین برای داشتن یک قضاوت عادلانه در مورد کیفیت مدلهای بر مبنای یادگیری ماشین باید تمام شرایط محیط بر مدل لحاظ گردد و تمرکز بر فقط یک معیار برای تصمیم‌گیری راجع به کیفیت مدل مناسب نیست.

به عنوان نتیجه‌گیری می‌توان گفت در این مطالعه یک مدل یادگیری انتقالیِ سبک از نظر محاسباتی که روی یک مجموعه تصویری کم حجم دارای ۱۱۲ تصویر از تصاویر مارهای غالب ایران آموزش دیده است، برای دسته‌بندی تصاویر در ۱۰ گروه استفاده شد. بر طبق نتایج به‌دست آمده هر چند عملکرد مدل عالی نبوده، اما به‌طور نسبی قابل قبول بوده است.

## منابع


Picek, L., Hrúz, M., Durso, A.M.& Bolon, I. 2022. Overview of SnakeCLEF 2022: automated snake species identification on a global scale, in CLEF 2022: conference and labs of the evaluation forum.

Amir, A., Zahri, N.A.H., Yaakob, N. & Ahmad, R.B. 2017. Image Classification for Snake Species Using Machine Learning Techniques," in Computational Intelligence in Information Systems. CIIS 2016. Advances in Intelligent Systems and Computing.

James, A. 2017. Snake classification from images. PeerJ Preprints.

Abdurrazaq, I.S., Suyanto, S. & Utama, D.Q. 2019, Image-Based Classification of Snake Species Using Convolutional Neural Network. 2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), pp 97-102.

Abayaratne, S., Ilmini, W. & Fernando, T. 2019. Identification of Snake Species Found in Sri Lanka Using Convolutional Neural Networks, in Conference: 3rd SLAAI - International Conference on Artificial Intelligence.

Progga, N., Rezoana, N.A., Hossain, M., Islam, R. & Andersson, K. 2021. A CNN Based Model for Venomous and Non-venomous Snake Classification, Applied Intelligence and Informatics, pp. 216-231.

Yang, Z. & Sinnott, R. 2021. Snake Detection and Classification using Deep Learning," in the 54th Hawaii. Int Conf Syst Sci.

Kalinathan, L., Balasundara, P., Ganesh, P., Bathala, S.S. & Mukesh, R.K. 2021. Automatic Snake Classification using Deep Learning Algorithm, in Conference and Labs of the Evaluation Forum.

Desingu, K., Palaniappan, M. & Kumar, J. 2021. Snake Species classification using Transfer learning," in CLEF 2021 – Conference and Labs of the Evaluation Forum.

۱- **پیوست**: کد مربوط به مدل

```
import torch
import torchvision
import matplotlib.pyplot as plt
from torch import nn
from torchvision import transforms
!pip install going_modular
import going_modular
try:
   from torchinfo import summary
except:
   print("[INFO] Couldn't find torchinfo... installing it.")
   !pip3 install -q torchinfo
   from torchinfo import summary
try:
   from going_modular.going_modular import data_setup, engine
except:
   print("[INFO] Couldn't find going_modular scripts... downloading them from GitHub.")
   !git clone https://github.com/mrdbourke/pytorch-deep-learning
   !mv pytorch-deep-learning/going_modular .
   !rm -rf pytorch-deep-learning
   from going_modular.going_modular import data_setup, engine
device = "cuda" if torch.cuda.is_available() else "cpu"
device
import os
import zipfile
from pathlib import Path
import requests
data_path = Path("data/")
image_path = data_path / "Iran_Snakes"
train_dir = image_path / "train"
test_dir = image_path / "test"
weights = torchvision.models.EfficientNet_B0_Weights.DEFAULT
auto_transforms = weights.transforms()
auto_transforms
train_dataloader, test_dataloader, class_names = data_setup.create_dataloaders(train_dir=train_dir,
                                                                               test_dir=test_dir,
                                                                               transform=auto_transforms,
                                                                               batch_size=32)
train_dataloader, test_dataloader, class_names
weights = torchvision.models.EfficientNet_B0_Weights.DEFAULT
model = torchvision.models.efficientnet_b0(weights=weights).to(device)
summary(model=model,
       input_size=(32, 3, 224, 224),
       col_names=["input_size", "output_size", "num_params", "trainable"],
       col_width=10,
       row_settings=["var_names"]
)
for param in model.features.parameters():
   param.requires_grad = False
torch.manual_seed(42)
torch.cuda.manual_seed(42)
output_shape = len(class_names)
model.classifier = torch.nn.Sequential(
   torch.nn.Dropout(p=0.2, inplace=True),
   torch.nn.Linear(in_features=1280,
                   out_features=output_shape,
                   bias=True)).to(device)
summary(model,
       input_size=(32, 3, 224, 224),
       verbose=0,
       col_names=["input_size", "output_size", "num_params", "trainable"],
       col_width=20,
       row_settings=["var_names"]
)
loss_fn = nn.CrossEntropyLoss()
```

```
optimizer = torch.optim.Adam(model.parameters(), lr=0.001)
torch.manual_seed(42)
torch.cuda.manual_seed(42)
from timeit import default_timer as timer
start_time = timer()
results = engine.train(model=model,
                       train_dataloader=train_dataloader,
                       test_dataloader=test_dataloader,
                       optimizer=optimizer,
                       loss_fn=loss_fn,
                       epochs=5,
                       device=device)
end_time = timer()
print(f"[INFO] Total training time: {end_time-start_time:.3f} seconds")
try:
    from helper_functions import plot_loss_curves
except:
    print("[INFO] Couldn't find helper_functions.py, downloading...")
    with open("helper_functions.py", "wb") as f:
        import requests
        request = requests.get("https://raw.githubusercontent.com/mrdbourke/pytorch-deep-learning/main/helper_functions.py")
        f.write(request.content)
    from helper_functions import plot_loss_curves
plot_loss_curves(results)
from typing import List, Tuple
from pathlib import Path
def get_folder(dataset):
    parent = Path(dataset).parent
    if "." in parent.name:
        return str(parent.parent)
    return str(parent)
from PIL import Image
def pred_and_plot_image(model: torch.nn.Module,
                        image_path: str,
                        class_names: List[str],
                        image_size: Tuple[int, int] = (224, 224),
                        transform: torchvision.transforms = None,
                        device: torch.device=device):
    img = Image.open(image_path)
    if transform is not None:
        image_transform = transform
    else:
        image_transform = transforms.Compose([
            transforms.Resize(image_size),
            transforms.ToTensor(),
            transforms.Normalize(mean=[0.485, 0.456, 0.406],
                                 std=[0.229, 0.224, 0.225]),
        ])
    model.to(device)
    model.eval()
    with torch.inference_mode():
      transformed_image = image_transform(img).unsqueeze(dim=0)
      target_image_pred = model(transformed_image.to(device))
    target_image_pred_probs = torch.softmax(target_image_pred, dim=1)
    target_image_pred_label = torch.argmax(target_image_pred_probs, dim=1)
    plt.figure()
    plt.imshow(img)
    plt.title(f"Pred: {class_names[target_image_pred_label]} | Prob: {target_image_pred_probs.max():.3f} | Real Parent
Directory: {get_folder(image_path)}")
    plt.axis(False);
import random
num_images_to_plot = 30
test_image_path_list = list(Path(test_dir).glob("*/*.jpg"))
test_image_path_sample = random.sample(population=test_image_path_list,
                                       k=num_images_to_plot)
for image_path in test_image_path_sample:
    pred_and_plot_image(model=model,
```

```
image_path=image_path,
class_names=class_names,
image_size=(224, 224))
```